Programming in C/C++
Exercises set five: grammatical parsers

Christiaan Steenkist
Jaime Betancor Valado
Remco Bos

February 20, 2017

Exercise 35, separated lists

We fixed the list grammar so it does what we want it to.

Design

To accomodate multiple datatypes we use listtokens instead of WORDs. To allow empty lists, lists can exist out of nothing (empty OR). Because a list with one item can’t be distinguished as a separated list we just count it as a normal list. A separated list is thus counted as `[listtoken ’,’ listtoken]` with optional repetition of `[listtoken ’,’]`. We abbreviated these nonterminals as `sepstart` and `sepend`. To simplify some more we defined `[listtoken ’,’]` as `septoken`.

Code listings

Listing 1: grammar.gr

```
%token WORD
%token INT
%token FLOAT

%%

list:
  plain
```

```
|
  separated
|
  // empty
;

plain:
  plain
  listtoken
|
  listtoken
;

separated:
  sepstart
  sepend
;

sepend:
  sepend
  septoken
|
  // empty
;

sepstart:
  listtoken
  septoken
;

septoken:
  ','
  listtoken
;

listtoken:
  WORD
|
  INT
|
  FLOAT
;
```